

ctools - Bug #2804

Inconsistencies between branch 2713-variability-tool and devel branch

01/24/2019 11:13 AM - Bonnefoy Simon

<b>Status:</b>	Closed	<b>Start date:</b>	01/24/2019
<b>Priority:</b>	Normal	<b>Due date:</b>	
<b>Assigned To:</b>	Bonnefoy Simon	<b>% Done:</b>	100%
<b>Category:</b>		<b>Estimated time:</b>	0.00 hour
<b>Target version:</b>	1.6.0		
<b>Description</b>			
<p>The branch 2713-variability-tool has been merged with the devel branch.</p> <p>However, it seems that there are some inconsistencies in the results between both branches for the ctfindvar tool.</p> <p>The devel branches exhibits variability with significance above 5 sigma in regions where no variability is expected (from simulation).</p> <p>Furthermore, it seems that running ctfindvar several times changes the results in the map.fits file.</p> <p>The dataset used is a simulation from ctobsim, with temporal dependency in the signal.</p> <p>Below is the lightcurve from branch 2713</p> <p>screenshot_1_1548324258_ctfindvar_2713_variability.png</p> <p>Below are shown different lightcurve from the devel branch.</p> <p>Each plots represents the output of one instance of ctfindvar on the same dataset.</p> <p>We can see that there are discrepancies between the two analysis.</p> <p>screenshot_2_1548324274_ctfindvar_devel_variability1.png</p> <p>screenshot_3_1548324281_ctfindvar_2713_variability.png</p>			

History

#1 - 01/24/2019 11:14 AM - Bonnefoy Simon

- File deleted (screenshot\_3\_1548324281\_ctfindvar\_2713\_variability.png)

#2 - 01/24/2019 11:17 AM - Bonnefoy Simon

- File ctfindvar\_devel\_variability2.png added

#3 - 01/24/2019 11:19 AM - Bonnefoy Simon

- File deleted (screenshot\_2\_1548324274\_ctfindvar\_devel\_variability1.png)

#4 - 01/24/2019 11:19 AM - Bonnefoy Simon

- File deleted (screenshot\_1\_1548324258\_ctfindvar\_2713\_variability.png)

#5 - 01/24/2019 11:20 AM - Bonnefoy Simon

- File ctfindvar\_devel\_variability1.png added

- File ctfindvar\_2713\_variability.png added

#6 - 02/11/2019 05:24 PM - Bonnefoy Simon

It seems that the bug comes from the multi process that is started on line 767 of ctfindvar.cpp, just before looping over all the pixels in the cube.

Commenting the line

```
//#pragma omp parallel for
```

seems to do the job.

#### #7 - 02/12/2019 05:53 PM - Knödlseider Jürgen

- Status changed from New to In Progress
- Target version set to 1.6.0
- % Done changed from 0 to 10

I looked into the code I could not find anything strange going on. Have you tested multi-processing before, and did this bug only occur after I integrated the code?

#### #8 - 02/12/2019 05:59 PM - Knödlseider Jürgen

You may add additional critical zones to the code in order to identify which of the instructions creates at the end a problem (if everything within the for loop is critical no multi-processing should happen, hence the bug should also disappear).

#### #9 - 02/13/2019 12:59 PM - Bonnefoy Simon

Yes, we tested multi-processing before and it was working well.  
It seems that the problems come from the line 777

```
GNDarray pixSig = get_variability_sig(ipix);
```

This function has been modified when merged to the devel branch.  
Before, the GNDarray in which the significance is stored for each pixel was given as an argument. Now it is created in the function.  
I don't know whether this might have an impact when multi-processing.

In the `ctfindvar::analyse_cube`, declaring the GNDarray `pixSig` at the beginning of the `analyse_cube` method, and adding a `pragma omp critical` before the `get_variability_sig` seems to be working fine.

#### #10 - 02/13/2019 02:03 PM - Knödlseider Jürgen

Maybe I have misunderstood what `get_variability_sig()` is doing. I thought that `get_variability_sig()` always returns an updated GNDarray, but looking again at the old code it seems that there can be conditions where the old values are kept:

```
// The GTI is discarded from bckg calculation and not checked again.  
if (!accepted_bin_bckg_vector[i]) {  
    continue;  
}  
// Check if bin fails minoff check or no observations overlap it  
else if (m_counts(pix_number, i) < m_minoff || alpha_vector[i] == 0) {  
    accepted_bin_bckg_vector[i] = false;  
    continue;  
}
```

If this is indeed the intention I would suggest to put back GNDarray as a method argument. In that case I suppose that no pragma is needed.

#### #11 - 02/14/2019 01:13 PM - Bonnefoy Simon

OK, it seems that the problem comes from the `ctfindvar_get_alphas` method.  
There was, in the branch 2713, a  
`#omp pragma critical(ctfindvar_get_alphas)`  
just before the  
`alphas[j] += exposure * bkg->rate_ebin(instdir, m_emin, m_emax);`  
line 1127.  
This critical is not in the devel branch.  
It seems that the problem is solved by just adding a critical here.

#### #12 - 02/15/2019 03:04 PM - Knödlseider Jürgen

- Status changed from *In Progress* to *Feedback*  
- % Done changed from 10 to 100

I added the line that you suggested to the code.

While writing up the code description for the H.E.S.S. paper, I also recognised that in `ctfindvar::get_variability_sig` the exclusion of bins with `m_counts(ipix, i) < m_minoff` was done in the iterative loop, which means that the bins were not necessarily excluded for the background computation. If I understood correctly your intention, the bins have to be excluded before the iterative loop is entered. I therefore modified the code as follows:

```
GNdarray ctfindvar::get_variability_sig(const int& ipix)
{
    // Initialise result
    GNdarray sig_histogram(m_gti.size());

    // Initialise vectors
    std::vector<bool> accepted_bin_bckg_vector(m_gti.size(), true);

    // Get alpha values for specified pixel
    std::vector<double> alphas = get_alphas(ipix);

    // Exclude all bins from the background estimate for which the
    // number of events is below "minoff" or for which alpha is zero
    for (int i = 0; i < m_gti.size(); ++i) {
        if (m_counts(ipix, i) < m_minoff || alphas[i] == 0.0) {
            accepted_bin_bckg_vector[i] = false;
        }
    }

    // Loop over pixels until all background pixels are removed
    bool background_validated = false;
    while (background_validated == false) {

        // Signal that background was validated
        background_validated = true;
    }
}
```

Can you please let me know if this is correct, and if the code now works for you as expected?

I integrated the code in the devel branch.

#13 - 02/15/2019 03:57 PM - Bonnefoy Simon

Yes, you are right, it is better to discard the GTIs with `m_counts(ipix, i) < m_minoff` before entering the while, so they don't participate in any way to the background calculation.  
I have tested the code. Everything seems to be working well from my side.  
Thank you very much!

#14 - 02/15/2019 04:31 PM - Knödlseeder Jürgen

- Status changed from Feedback to Closed

Great, I close then the issue.

Files

ctfindvar_devel_variability2.png	186 KB	01/24/2019	Bonnefoy Simon
ctfindvar_devel_variability1.png	215 KB	01/24/2019	Bonnefoy Simon
ctfindvar_2713_variability.png	211 KB	01/24/2019	Bonnefoy Simon